Alan Turing and Human-Like Intelligence

Peter Millican, Hertford College, Oxford

The concept of Human-Like Computing became central to visions of Artificial Intelligence through the work of Alan Turing, whose model of computation (1936) is explicated in terms of the potential operations of a human "computer", and whose famous test for intelligent machinery (1950) is based on indistinguishability from human verbal behaviour. But here I shall challenge the apparent human-centredness of the 1936 model (now known as the Turing machine), and suggest a different genesis of the idea with a primary focus on the foundations of mathematics, and with human comparisons making an entrance only in retrospective justification of the model. It will also turn out, more surprisingly, that the 1950 account of intelligence is ultimately far less human-centred than it initially appears to be, because the *universality* of computation – as established in the 1936 paper – makes human intelligence just one variety amongst many. It is only when Turing considers *consciousness* that he treats intelligence in a way that cannot properly be carried over to machines. But here he is mistaken, since his own work gave ample reason to reinterpret intelligence as sophisticated information processing for some purpose, and to divorce this from the subjective consciousness with which it is humanly associated.

The Background to Turing's 1936 Paper

Alan Turing's remarkable 1936 paper, "On Computable Numbers, with an Application to the *Entscheidungsproblem*", introduced the first model of an all-purpose, programmable digital computer, now universally known as the *Turing machine*. And the paper, as noted above, gives the impression that this model is inspired by considering the potential operations of a human "computer". Yet the title and organisation of the paper suggest instead that Turing is approaching the topic from the direction of fundamental issues in the theory of mathematics, rather than any abstract analysis of human capabilities. It will be useful to start with an overview of two essential components of this theoretical background.

The first of these components is Georg Cantor's pioneering work on the *countability* or *enumerability* of various infinite sets of numbers: the question of whether the elements of these sets could in principle be set out – or *enumerated* – in a single list that contains every element of the set at least once. Cantor had shown in 1891 that such enumeration of *rational* numbers (i.e. fractions of integers) is indeed possible, since they can be exhaustively ordered by the combined magnitude of their numerator and denominator.[1] *Real* numbers, however, cannot be enumerated, as demonstrated by his celebrated *diagonal* proof, which proceeds by *reductio ad absurdum*. Focusing on real numbers between 0 and 1 expressed as infinite decimals,[2] we start by assuming that an enumeration of these *is* possible, and imagine them laid out accordingly in an infinite list R (so we are faced with an array which is infinite both horizontally, owing to the infinite decimals, and vertically, owing to the infinite list). We then imagine constructing another infinite decimal α by taking its first digit $\alpha[1]$ from the first real number in the list r_1 (so $\alpha[1] = r_1[1]$), its second digit from the second real number in the list ($\alpha[2] = r_2[2]$), its third digit from the third real number in the list ($\alpha[3] = r_3[3]$), and so on. Thus α is the infinite decimal that we get by tracing down the diagonal of our imagined array: in every case α has its n^{th} digit in common with r_n. We now imagine constructing another infinite decimal number β from α, by systematically changing every single digit according to some rule (e.g. if $\alpha[n] = 0$, then $\beta[n] = 1$, else $\beta[n] = 0$). For any would-be enumeration R, this gives a systematic method of constructing a number β whose n^{th} digit $\beta[n]$ must in every case be *different* from the n^{th} digit of r_n. Thus β cannot be identical with any number in the list, contradicting our assumption that R was a complete enumeration, and it follows that no such enumeration is possible.

The second essential component in the background of Turing's paper is David Hilbert's *decision problem* or *Entscheidungsproblem*: can a precise general procedure be devised which is able, in finite time and using finite resources, to establish whether any given formula of first-order predicate logic is provable or not? A major goal of Hilbert's influential programme in the philosophy of mathematics was to show that such *decidability* was achievable, and his 1928 book with Wilhelm Ackermann even declared that "The decision problem must be called the main problem of mathematical logic" (p. 77). The problem accordingly featured prominently in Max Newman's Cambridge University course on the Foundations of Mathematics, attended by Alan Turing in spring 1935. But by then Gödel's incompleteness theorems of 1931 – also covered in Newman's course – had shown that two other major goals of Hilbert's programme (proofs of *consistency* and *completeness*) could not both be achieved, and Turing's great paper of 1936 would show that decidability also was unachievable.

A potentially crucial requirement in tackling the *Entscheidungsproblem* – especially if a negative answer is to be given to the question of decidability – is to pin down exactly what types

[1] For example, 1/1 (sum 2); 1/2, 2/1 (sum 3); 1/3, 2/2, 3/1 (sum 4); 1/4, 2/3, 3/2, 4/1 (sum 5); and so on – every possible fraction of positive integers will appear somewhere in this list. To include negative fractions of integers, we could simply insert each negative value immediately after its positive twin.

[2] Or alternatively *binimals*, as discussed below.

of operation are permitted within the would-be general decision procedure. For without some such circumscription of what is permissible (e.g. ruling out appeal to an all-knowing oracle or deity, if such were to exist), it is hard to see much prospect of delimiting the range of what can theoretically be achieved. An appropriate limit should clearly prohibit operations that rely on unexplained magic, inspiration, or external intervention, and should include only those that are precisely specifiable, performable by rigorously following specific instructions in a "mechanical" manner, and reliably yielding the same result given the same inputs. A procedure defined in these terms is commonly called an *effective method*, and results thus achievable are called *effectively computable*. These concepts remain so far rather vague and intuitive, but what Turing does with the "computing machines" that he introduces in §1 of his 1936 paper is to define a precise concept of effective computability in terms of what can be achieved by a specific kind of machine whose behaviour is explicitly and completely determined by a lookup table of conditions and actions. Different tables give rise to different behaviour, but the scope of possible conditions and actions is circumscribed precisely by the limits that Turing lays down.

Introducing Turing Machines

As its title suggests, the 1936 paper starts from the concept of a *computable* number:

> "The 'computable' numbers may be described briefly as the real numbers whose expressions as a decimal are calculable by finite means." (p. 58)

Turing's terminology is potentially confusing here, in view of what follows. The numerical expressions he will actually be concerned with express real numbers *between 0 and 1*, interpreted in *binary* rather than *decimal* (i.e. sequences of "0" and "1", following an implicit *binary point*, as opposed to sequences of decimal digits following a decimal point). To provide a distinctive term for such binary fractions, let us call them *binimals*. For example, the first specific example that Turing gives (§3, p. 61)[3] generates the infinite sequence of binary digits:

0 1 0 1 0 1 0 1 0 1 …

which is to be understood as expressing the recurring binimal fraction $0.\overline{01}$, numerically equivalent to 1/3.[4] That the binimal recurs to infinity is no difficulty: on the contrary, *all* of

[3] In what follows, references of this form, citing section and page numbers, are always either to the 1936 paper or – later – to the 1950 paper. Note also that for convenience, all page references to Turing's publications are to the relevant reprint in Copeland (2004).

[4] The "1" in the second binimal place represents 1/4, and the value of each subsequent "1" is 1/4 of the previous one. So we have a geometric series 1/4 + 1/16 + 1/64 + … whose first term is 1/4 and common ratio 1/4, yielding a sum of 1/3 by the familiar formula *a*/(1-*r*).

Turing's binimal expressions will continue to infinity, whether recurring (as in the binimal for 1/2, i.e. 0.1000 …) or not (as in the binimal for π/4).[5]

That Turing's binimal expressions continue to infinity may well seem surprising, given his declared aim to explore those that are *computable*, glossed as "calculable by finite means". At the very beginning of §1 of his paper he acknowledges that the term "requires rather more explicit definition", referring forward to §9 and then commenting "For the present I shall only say that the justification lies in the fact that the human memory is necessarily limited." (p. 59). In the next paragraph he compares "a man in the process of computing a real number to a machine which is only capable of a finite number of conditions". So the finitude he intends in the notion of *computable numbers* does not apply to the ultimate extent of the written binimal number, nor therefore to the potentially infinite *tape* – divided into an endless horizontal sequence of *squares* – on which the digits of that number (as well as intermediate workings) are to be printed. Rather, what is crucial to Turing's concept of computability "by finite means" is that the choice of behaviour at each stage of computation is tightly defined by a finite set of machine memory *states*,[6] a finite set of *symbol types*, and a limited range of resulting *actions*. Each possible combination of state and symbol – the latter being read from the particular square on the tape that is currently being *scanned* – is assigned a specific repertoire of actions. Computation takes place through a repeated sequence of scanning the current square on the tape, identifying any symbol (at most one per square) that it contains, then performing the action(s) assigned to the relevant combination of current state and symbol.

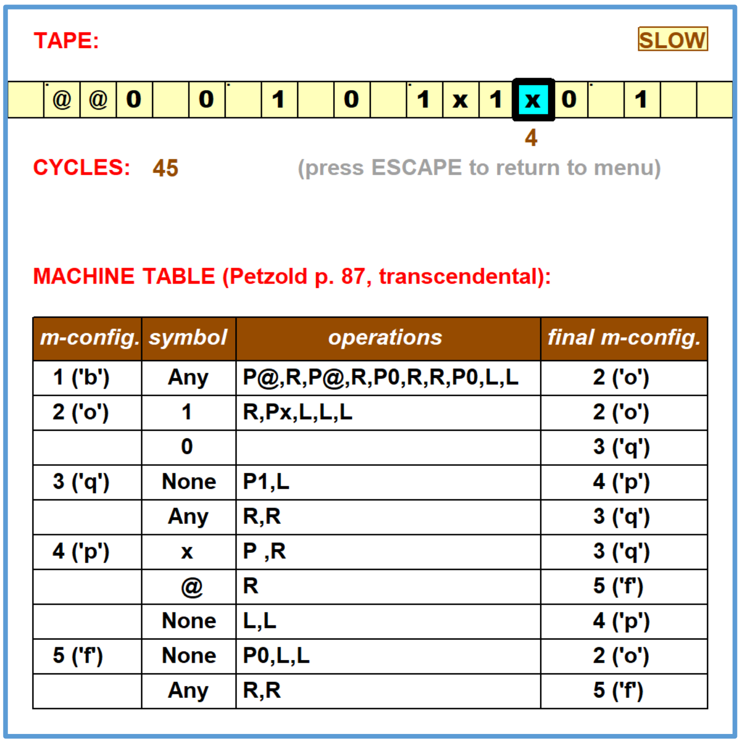
For theoretical purposes, the actions for each state/symbol combination are very tightly constrained, limited to printing a symbol (or blank) on the current square, moving the scanner one square left or right, and changing the current state. For much of his paper, however, Turing slightly relaxes these constraints, allowing multiple printings and movements, as in the example illustrated below. This shows the machine defined by a table specified in §3 of the 1936 paper (p. 62), running within a Turing machine simulator program.[7] Starting with an empty tape in state 1 (or "b" in Turing's paper), the machine prints the sequence of symbols we see at the left of the tape ("P@" prints "@"; "R" moves right; "P0" prints"0"), then moves left twice ("L,L") to return to the square containing the first "0", before transitioning into state 2. Next, finding

[5] Note that a decimal or binimal will recur if, and only if, it represents a *rational* number, i.e. a fraction of integers.

[6] Turing's term for what is now generally called a *state* is an "*m*-configuration" (§1, p. 59).

[7] By far the best way of familiarising oneself with the operation of Turing machines is to see them in action. The Turing machine simulator illustrated here is one of the example programs built into the *Turtle System*, freely downloadable from www.turtle.ox.ac.uk. Within the simulator program, this particular machine table is available from the initial menu, which also includes some other relevant examples taken from Petzold's excellent (2008).

itself now scanning an “0” in state 2, it transitions into state 3. Next, still scanning an “0” but now in state 3, it moves right twice and stays in state 3. Then, scanning a blank square (“None”) in state 3, it prints a “1” and moves left, transitioning into state 4. And so on.



TAPE:

	@	@	0		0		1		0		1	x	1	x	0		1		
														4					

CYCLES: 45

(press ESCAPE to return to menu)

MACHINE TABLE (Petzold p. 87, transcendental):

m-config.	symbol	operations	final m-config.
1 ('b')	Any	P@,R,P@,R,P0,R,R,P0,L,L	2 ('o')
2 ('o')	1	R,Px,L,L,L	2 ('o')
	0		3 ('q')
3 ('q')	None	P1,L	4 ('p')
	Any	R,R	3 ('q')
4 ('p')	x	P ,R	3 ('q')
	@	R	5 ('f')
	None	L,L	4 ('p')
5 ('f')	None	P0,L,L	2 ('o')
	Any	R,R	5 ('f')

A Turing Machine in Action

Turing’s machine table has been cleverly designed to print out an infinite binimal sequence, which has the interesting property of never recurring:

0 0 1 0 1 1 0 1 1 1 0 1 1 1 1 0 1 1 1 1 1 …

Adding an initial binimal point, this yields a binimal number which is clearly *irrational* (precisely because it never recurs), but which has been shown to be *computable* in the sense that there exists a Turing machine which generates it, digit by digit, indefinitely.[8] This notion is trivially extendable to binimals that also have digits before the binimal point. And in this extended sense it is clear – given that numerical division can be mechanised on a Turing machine – that all rational numbers are computable, so it now follows that *the computable numbers are strictly more extensive than the rational numbers*. Indeed, Turing will go on later, in §10 of the paper, to explain that all of the familiar irrational numbers, such as π, e, $\sqrt{2}$, and indeed all real roots of rational algebraic equations, are computable.

[8] To avoid reference to the infinite sequence of digits, we can formally define a binimal number as computable if and only if there exists a Turing machine such that, for all positive integers *n*, there is some point during the execution of that machine at which the tape will display the subsequence consisting of its first *n* digits.

The Fundamental Ideas of Turing's 1936 Paper

Accepting for the moment Turing's key claim that the scope of what is calculable by his computing machines does indeed coincide with the desired notion of *computability*, we can readily see how the work of Cantor could have provided clear inspiration for his subsequent method of proceeding. To start with, all these machines are by definition entirely deterministic,[9] so each can compute, in binimal form, at most one computable number. They succeed in computing such a number if and only if – when left to proceed indefinitely – they would continue to generate a potentially infinite sequence of binary digits on the tape. Turing calls these machines "circle-free" (in contrast to a "circular" machine that stops generating binary digits, for example by getting into a non-printing loop). But then, since – by Turing's key claim – any computable number must correspond to at least one machine table that is capable of computing it, there cannot be more computable numbers than there are circle-free machine tables.[10]

With his thinking apparently deeply informed by Cantor's methods, it seems plausible that Turing at this point would have quite quickly seen the potential for paradox. For on the one hand, since his machine tables can straightforwardly be reduced to a linear textual form, while lines of text can be ordered by length and then lexicographically (i.e. quasi-alphabetically, acknowledging both letters and other symbols), it immediately follows that machine tables, and hence the numbers they are capable of generating, must be enumerable – we simply list the numbers according to the ordering of the tables that generate them.[11] But on the other hand, since this list *C* of computable binimals will produce exactly the kind of infinite array from which Cantor's diagonal argument arose, it is very natural to wonder whether a similar diagonal argument would work here, to demonstrate by *reductio ad absurdum* that the supposed infinite list of computable numbers *C* *cannot* be a complete enumeration. A puzzling contradiction seems to be in prospect.

Following this thread leads directly to one of the great innovations in Turing's paper – the Universal Turing Machine. A diagonal argument can generate a paradox here only if the application of that argument is *computable*, for there is no contradiction in defining a real number that is not on list *C*, if that number is not itself computable. Hence the obvious next step is to explore how one might attempt to compute the paradoxical number, which Turing calls β (§8,

[9] Turing explicitly states that his paper considers only "automatic machines" (§2, p. 60).

[10] In fact any computable number can be generated by an infinite number of machine tables, since arbitrary irrelevant states can be added without affecting the behaviour (see §5, p. 68).

[11] Note that it does not matter if computable numbers occur in the list more than once, as long as all are present.

p. 72). This requires (a) iterating through all possible machine tables in some appropriate order; (b) identifying in turn those that are circle-free; then (c) generating the n^{th} digit from the n^{th} circle-free table and adjusting it according to the same rule that we used in the case of Cantor's argument (i.e. if $c_n[n] = 0$, then $\beta[n] = 1$, else $\beta[n] = 0$). If these three steps are all achievable by computational methods, then we shall have a genuine paradox: β will be computable, and yet absent from the complete enumeration of computable numbers.

Of the three stages involved here, the most straightforward is to iterate through all possible machine tables. As noted earlier, any machine table can be reduced to a linear textual form. Then the individual symbols that occur in this linear text can be systematically translated as decimal digits (or sequences of digits), thus reducing the entire table to a (very large) integer. Translation in the reverse direction is also fairly easy, so it is in principle unproblematic to iterate up through the positive integers, identifying those that are potential "description numbers" of Turing machine tables.[12]

Having thus mechanised the identification of each possible table in turn, we now need to be able to generate the n^{th} digit from the n^{th} relevant table, and this is where we require a Universal Turing Machine: a machine which can simulate the operation of any given machine table so as to generate that n^{th} digit.[13] Turing's proof that such a universal machine is possible (in §§6-7, pp. 68-72 of his paper) was an intellectual *tour de force* and a landmark in the theory of computation, proving for the first time the possibility of a universal programmable computer. But envisaging the *possibility* of such a machine did not require any comparable effort of imagination: as we have seen, it arose naturally from the Cantorian context of his investigation into computable numbers.

Stages (a) and (c) of the paradox-generating process have turned out to be achievable; hence if paradox is to be avoided, it must fail at stage (b), which involves identifying whether a given machine table is circle-free. Turing himself draws this conclusion early in §8 of his paper (p. 72), but he also acknowledges that this indirect argument might leave readers unsatisfied, and he accordingly goes on to explore in more detail how the generation of β is bound to fail. The crucial difficulty arises when the hypothetical checking machine – which has been designed to

[12] Turing explains description numbers in §5 of his paper, entitled "Enumeration of computable sequences" (pp. 66-8). The idea of translating complex formulae – and even sequences of formulae – into single large integers (and back again) would already have been very familiar to Turing through his study of Gödel's theorems.

[13] Note that Turing's initial machines, such as the one illustrated in the image of the simulator above, all start off with a completely blank tape, and then generate the relevant binimal number rightwards along the tape from the starting position. The Universal Machine, however, begins with a tape that already contains, to the left, an encoded version of the machine table that is to be simulated. The Universal Machine then performs its simulation by constantly referring back to that encoded table to work out what needs to be done at each stage.

test, in turn, whether each of the enumerated machine tables is circle-free – comes to test its own machine table (p. 73). This is essentially the same point that is now very familiar from proofs of the unsolvability of the Halting Problem: the supposed halting oracle has to fail when it comes to test (a slightly modified version of) itself.[14]

Turing has now identified something that is in general uncomputable: whether an arbitrary machine table is, or is not, circle-free. And from this point in the paper he accordingly changes focus from the domain of Cantor – enumerability of computable numbers and diagonalisation – to the domain of Hilbert and his decision problem. The "application to the *Entscheidungsproblem*" promised by Turing's title requires him to translate his result about the uncomputability of machine behaviour into some parallel result about the undecidability of formulae in predicate logic. As a first step he neatly proves an important lemma, that since there is no computable test for whether an arbitrary machine is circle-free, nor can there be a computable test for whether an arbitrary machine will ever print a given symbol such as "0" (§8, pp. 73-4). The remainder of the task, which he postpones to §11, is technically tricky but conceptually quite straightforward. First, having defined appropriate predicates, he shows how, given any machine *M*, it is possible to construct a predicate formula Un(*M*) which states, in effect, *that M will at some point print "0"* (pp. 84-5). He goes on to show that Un(*M*) will be *provable* if and only if *M does indeed* at some point print "0". But then, since there is no "general (mechanical) process" for determining whether *M* ever prints "0", it follows that there can be no such "general (mechanical) process for determining whether Un(*M*) is provable". "Hence the Entscheidungsproblem cannot be solved" (p. 87).

Justifying the Turing Machine

Sandwiched between §8, where Turing proves his important lemma, and §11, where he applies it to the *Entscheidungsproblem*, are two sections aiming to justify the adequacy of his model of computation, by showing "that the 'computable' numbers [in the technical sense defined by his machines] include all numbers which would naturally be regarded as computable" (§9, p. 74). Turing offers arguments "of three kinds", the first of which – described as "A direct appeal to intuition" (p. 75) – gives the best evidence that he is basing his machines on an abstraction from

[14] Suppose – for *reductio* – that *H(P,T)* is a program that infallibly tests whether any arbitrary program *P* will halt given input *T*, outputting 'Yes' or 'No' accordingly (and then halting). We can create a paradoxical program *K* from *H*, by replacing "print('Yes')" with a non-terminating loop such as "repeat print('Yes') until 0=1", for then a positive halting verdict will fail to halt, while a negative halting verdict still halts. And hence the result of *K(K,K)* cannot consistently be assigned: it should halt if and only if it doesn't halt. Thus the supposition that program *H* exists leads to a contradiction: there can be no such program.

the operations of a human "computer". Now he expands on ideas briefly sketched in §1 (p. 59), where we saw that he appeals to the finitude of human memory as justification for insisting on "finite means" of calculability. Here in §9 (pp. 75-7) he gives a far more elaborate argument, to the effect that his tape-based machines can, in principle, mimic any kind of systematic calculation that can be done by a human "computer". He argues in turn that a tape divided into squares provides an appropriate simplification of a human notebook; that there should be a bound to the number of squares that can be observed at any one moment; and that the symbols used in calculation and the number of possible "states of mind" must be finite in number, reflecting our limited recognitional capacities, because "If we were to allow an infinity of symbols … [or] an infinity of states of mind, some of them will be 'arbitrarily close' and will be confused" (pp. 75, 76). He then goes on to explain how "the operations performed by the [human] computer" can plausibly be split up into "simple operations", each of which "consists of some [elementary] change of the physical system consisting of the computer and his tape", involving no more than one change of symbol, a move to observe an adjacent square, or a change of state of mind. Thus we reach the design of the Turing machine, apparently starting from an abstract analysis of how a human "computer" operates.

Turing's second kind of argument for the adequacy of his model of computation involves showing the equivalence of the resulting notion of computability with that of other established definitions. In the remainder of §9 he explains how a Turing machine can be constructed "which will find all the provable formulae" of a systematised version of "the Hilbert functional calculus", or what we now know as first-order predicate logic (pp. 77-8). In the Appendix to the paper (pp. 88-90), he goes on to prove, in outline, that his notion of computability also coincides with Alonzo Church's concept of effective calculability (or λ-definability).

Turing's third kind of argument occupies §10 of the paper, appropriately entitled "Examples of large classes of numbers which are computable" (pp. 79-83). These classes encompass various combinations and iterations of computable functions, the root of any computable function that crosses zero, the limit of any "computably convergent sequence", π, *e*, and all real algebraic numbers. After all this, the reader is left with an appreciation of the comprehensive power of Turing machines, rendering plausible the claim that they can indeed circumscribe the appropriate boundaries of our intuitive notion of effective computability. Indeed this claim, that the functions that are effectively computable are to be identified with those that are Turing-machine computable – or equivalently, λ-definable (Church) or general recursive (Gödel) – is now known as the *Church-Turing thesis*, and widely accepted on the basis of Turing's arguments.

Was the Turing Machine Inspired by Human Computation?

Andrew Hodges' magnificent 1983 biography, which helped to bring Turing to public prominence, describes the early pages of §9 of the 1936 paper as "among the most unusual ever offered in a mathematical paper, in which [Turing] justified the definition [of a computing machine] by considering what *people* could possibly be doing when they 'computed' a number …" (p. 104). According to Mark Sprevak, "A Turing machine is an abstract mathematical model of a human clerk. … Turing wanted to know which mathematical tasks could and could not be performed by a human clerk." (2017, p. 281). And beyond mere modelling, Robin Gandy credits Turing with having *proved* a theorem of similarity between human and machine computability: "He … considers the actions of an abstract human being who is making a calculation … [and] the limitations of our sensory and mental apparatus. … Turing easily shows that the behavior of the computor can be exactly simulated by a Turing machine. … [His] analysis … *proves a theorem* … *Any function which can be calculated by a human being following a fixed routine is computable*." (1988, pp. 81-3).

Turing's later publications about machine intelligence – which we shall come to shortly – make it especially tempting to see the Turing machine as inspired by human computation and the desire to model it. Hodges accordingly considers it "striking … that the Turing machine, formulated 14 years before the 'Turing Test', was also based on a principle of imitation. The 'machine' was modelled by considering what a human being could do when following a definite method." (2009, p. 14). This can suggest a unity of purpose linking the mathematical analysis of the 1936 paper to the more philosophical discussions that would emerge over a decade later:

> "The problem of mind is the key to 'Computable Numbers'. … [Turing] wished from the beginning to promote and exploit the thesis that all mental processes – not just the processes which could be explicitly described by 'notes of instructions' – could be faithfully emulated by logical machinery" (Hodges 1988, pp. 6, 8).

Hodges here goes further than most commentators on Turing's paper, but many have found it plausible that modelling the limits of algorithmic *human* thinking was the principal driver behind the design of the Turing machine, even if Turing's main aim in doing so was not to provide a model of human thinking for its own sake, but rather to circumscribe the notion of an effective method on the way to tackling the *Entscheidungsproblem*. Gandy apparently takes this view: "I suppose, but do not know, that Turing, right from the start of his work, had as his goal a proof of the undecidability of the Entscheidungsproblem" (1988, p. 82).

Against all this, I would like to suggest that Turing's design of his computing machines was not primarily intended to model human thinking, but nor did it derive primarily from his desire to resolve the *Entscheidungsproblem*. Instead, I believe that Turing's enquiry – and his conception of computing machines – started from an interest in the notion of *computable numbers* themselves. Having devised his machines and appreciated their power, he then saw the need to provide some argument for their theoretical generality, but it was probably quite late in the process that Turing turned to doing this in any detail (in §9 of his paper, as we have seen). Moreover their "application to the *Entscheidungsproblem*" was not, I suspect, foreseen in advance, but became apparent during the course of his investigation into computable numbers.

This interpretative hypothesis derives obvious support from the title of Turing's paper, and from its ordering and organisation (bearing in mind that it was written before the days of wordprocessors, when restructuring and reordering could be tedious in the extreme). But more substantially, there is a clue to the origin of Turing's thought in a footnote reference that has been generally overlooked, but which coheres extremely well with the explanation above of how the fundamental ideas of the 1936 paper naturally emerge once an investigation into computable numbers is under way.

The footnote in question is attached to the end of the first sentence of §8 (entitled "Application of the diagonal process"):

> "It may be thought that arguments which prove that the real numbers are not enumerable would also prove that the computable numbers and sequences cannot be enumerable.[4]" (p. 72)

Footnote 4 itself reads "*Cf.* Hobson, *Theory of functions of a real variable* (2nd ed., 1921), 87, 88." E. W. Hobson was G. H. Hardy's immediate predecessor as Sadleirian Professor of Pure Mathematics at Cambridge, retiring in 1931. His *Theory of Functions* was the first English book covering Lebesgue integration and measure, and was highly influential on teaching practice, so it should be no surprise that Turing, who arrived at Cambridge as an undergraduate in Mathematics that same year, was familiar with it.[15]

On following up this reference, one might naturally expect to find an exposition of Cantor's familiar diagonal argument.[16] But that argument occurs in Hobson's book at pages 82-3, and the discussion over pages 87-8 is quite different. Here Hobson is concerned with the

[15] I am grateful to Patricia McGuire, Archivist of King's College, for ascertaining that Hobson's second edition was acquired by the college library on January 28 1921.

[16] Indeed, my own primary interest in consulting it was to discover whether Turing had encountered the Cantorian argument as expressed in binary or decimal form – Hobson uses decimals. It was a delightful surprise to find, instead of Cantor's argument, clear corroboration for my interpretative hypothesis.

definability of numbers, particularly in the light of paradoxes such as those published in 1905 by Julius König and Jules Richard. This discussion in Hobson's book draws on papers that he had presented to the London Mathematical Society in January and November 1905 (the latter in response to König's September publication). The initial paper quickly stimulated several responses in the Society's *Proceedings,* from G. H. Hardy, A. C. Dixon, Bertrand Russell, and Philip Jourdain.[17] Moreover it is mentioned (somewhat approvingly), together with König's paper and Dixon's response, in a footnote to Whitehead and Russell's *Principia Mathematica*.[18]

Hobson's 1921 discussion of *definable* numbers bears striking parallels with Turing's 1936 discussion of *computable* numbers. Hobson emphasises repeatedly that finite definitions from any "given stock of words and symbols" must be enumerable, and alludes (albeit disapprovingly) to König's inference that real numbers "fall into two classes … [those] capable of finite definition, and those … inherently incapable of finite definition" (p. 87). He also draws explicit attention to Cantor's diagonal argument as a method of defining further numbers that were not in the original enumerable set, and although he himself avoids any suggestion of paradox here, he does mention both König and Richard in a footnote (along with his own initial 1905 paper, and *Principia Mathematica*). Hobson takes diagonalisation to show "that there exists, and can exist, at any time, no stock of words and symbols which cannot be increased for the purpose of defining new elements of the continuum", but he denies "that there exists any element of the continuum that is inherently incapable of finite definition" (pp. 87, 88).[19]

Hobson's discussion is also somewhat suggestive of another prominent theme in Turing's paper, that of defining or computing a number by constructing it digit-by-digit. He couches Cantor's argument in terms of "a set of rules by means of which the *m*th digit of the *n*th number [in an enumeration] can be determined" (p. 82). And he provides a "finite definition" of any number that is the limit of a convergent sequence $\{x_n\}$, in terms of successive m^{th} digits each of which is "defined as that digit which is identical with the m^{th} digit of an infinite number of the elements" of the sequence (p. 88).

[17] All, incidentally, Cambridge men. Hardy and Russell, at least, were well known to Turing; Jourdain died in 1919; Dixon retired in 1930 (from Queen's University Belfast to Middlesex), but from 1931 until 1933 was President of the London Mathematical Society, which would later publish Turing's 1936 paper.

[18] In the second edition of 1927, this is at p. 61 of Volume 1. The footnote is to a numbered paragraph explaining König's paradox, and immediately followed by another explaining Richard's paradox. For discussion, and a reprint of Russell's response to Hobson, see Russell's *Collected Papers, Volume 5* (2014), chapter 2.

[19] Hobson's mathematical argument on p. 88 appears to take for granted, without any obvious basis, that a sequence of definable elements must itself be definable (and hence that its limit must also be definable). Fan (2020) explicates Hobson's thinking in terms of the requirement for a "*norm* by which [an] aggregate is defined" (p. 131); see also §5 (pp. 135-7) on "Hobson's Way Out" and §6 (pp. 137-8) which includes some remarks on Turing.

There is more than enough here to give a plausible account of the initial inspiration for Turing's landmark paper. Hobson starts by raising the question of whether there is a tenable distinction between *definable* and *non-definable* numbers. But Turing – perhaps in the wake of Newman's lecture course with its focus on effective "mechanical" methods, and perhaps in order to pin the problem down in terms of concrete calculation of numbers digit-by-digit (as just discussed) – now interprets *definition* in terms of *calculability*.[20] He then designs a calculating machine whose purpose is precisely to generate numbers on an endless tape, something very different from what any human "computer" or "clerk" is likely to be doing, but exactly on target in Hobson's context.[21] Next comes the idea of enumerating the possible machines, a close parallel to the enumeration of definitions that is key to the paradoxes of König and Richard. The parallel to the Richard paradox runs even deeper, since this involves explicit diagonalisation, identifying the n^{th} digit of the n^{th} legitimate definition, just as Turing's diagonal argument – applied in the paragraph immediately following the Hobson footnote – involves identifying the n^{th} digit generated by the n^{th} circle-free machine.[22] Both diagonal processes, moreover, crucially raise the issue of testing for legitimacy: in Richard's case, whether an enumerated string is a genuine number definition; in Turing's case, whether a machine description number is genuinely that of a circle-free machine. But this last issue would presumably have occurred to Turing even without any prior example, for as we saw, it follows directly from the logic of his argument – on pain of paradox – that whether a machine is circle-free has in general to be uncomputable.

Understanding Turing's process of thought as arising from a concern with *definable* numbers coheres well with the structure of his paper, and also fits naturally into what we know of his intellectual context. The outline above also explains why Turing was able to work so fast as to astonish Newman by presenting him with the completed draft of his paper in April 1936

[20] Turing's diagonal process, however, ultimately forces a distinction between the two notions, as anticipated in the second paragraph of his paper: "The computable numbers do not … include all definable numbers, and an example is given of a definable number [β as described earlier] which is not computable" (1936, p. 58). I suspect that recognition of this came *after* Turing had pursued the potential paradox that arises from identification of the two notions, but in any case, this early prominence of the issue within the finished paper somewhat corroborates my hypothesis that he started from an interest in *definable* numbers.

[21] Hodges (2013) suggests another possible inspiration for this kind of machine, namely, Turing's interest in "normal numbers", on which his King's College friend David Champernowne had published a paper (1933), and the topic of a note that Turing himself wrote soon after submitting the 1936 paper (drafted on the back of its typescript). A normal number is one whose digits and groups of digits are all uniformly distributed in the infinite limit.

[22] Turing had a deep interest in philosophy of mathematics, even extending to giving a Moral Sciences Club paper on the subject in December 1933 (see Hodges 1983, pp. 85-6). So quite independently of Hobson's textbook, he would already have known about Richard's paradox, whose salience was very clear to those working on the foundations of mathematics in the 1930s. For example, the paradox is mentioned in *Principia Mathematica* (see note 18 above); Gödel's famous paper of 1931 remarks: "The analogy between this result and Richard's antinomy leaps to the eye" (1962, p. 40); and Alonzo Church's (1934) is devoted to the paradox.

(Hodges 1988, p. 3). Turing later told Gandy "that the 'main idea' of the paper came to him when he was lying in Grantchester meadows in the summer of 1935" (Gandy 1988, p. 82). This suggests that there was one leading idea that crystallised everything in Turing's mind, rather than a succession of original points.[23] The hypothesis sketched above is entirely consistent with this, because it builds so much of his argument from materials that were already familiar to him. If that hypothesis is correct, then Turing's "main idea" in Grantchester meadows might have been his initial recognition that there was potential for a paradox regarding computability, structurally parallel to the familiar paradoxes of definability. That, however, was a relatively small step in the context, simply linking Hobson's discussion (and/or Richard's paradox) with the idea of "mechanical" computability emphasised in Newman's lectures of spring 1935. Far more substantial inspiration would have been required for Turing to come up with his entirely innovative design for a tape-based and table-driven computing machine capable of generating endless binimals: that design, I strongly suspect, was his "main idea".

From 1936 to 1950

In the decade following his 1936 paper, Turing experienced the most dramatic confirmation of his confidence in the powers of automated computation, working for much of that time at Bletchley Park cracking Nazi codes, to the huge benefit of the allied war effort. After the war, he was recruited by the National Physical Laboratory to design the Automatic Computing Engine (ACE), of which an initial version (Pilot ACE) would be operational by 1950. But by May 1948, he was instead employed at Manchester University, having been repelled by institutional politics and hiatus at the NPL, and attracted by the offer of a job from Newman, who had in 1945 been appointed Head of the Manchester Mathematics Department. This gave Turing, as Deputy Director of the Computing Machine Laboratory, the opportunity to work on the world's first electronic stored-program digital computer, the "Manchester Baby", and subsequently on the first such computer to be manufactured commercially, the Ferranti Mark I. Now he had the practical prospect of indulging his intense interest in machine intelligence, notably by developing – with his college friend David Champernowne – a chess-playing program called *Turochamp* (though the Ferranti proved insufficiently powerful to run the completed program).[24]

[23] Gandy continues: "The 'main idea' might have been either his analysis of computation, or his realization that there was a universal machine, and so a diagonal argument to prove unsolvability." (1988, p. 82). But the latter seems unlikely, for we have seen that diagonalisation was a standard part of the mathematical repertoire, while the need for a universal machine – though admittedly not the method of implementing it – is fairly easy to appreciate once the requirement of mechanising a diagonal computing process has been noticed.

[24] There is now a wealth of material on Turing's history over this period. See for example Hodges (1983), pp. 314-415; Copeland (2004), pp. 353-77, 395-401; Copeland *et al.* (2017), pp. 199-221.

Perhaps more than anyone else in the world at this time, Turing was aware of the amazing potential of digital computers. The theoretical limits that he had identified in 1936 were of little practical consequence now; the far more significant result was that even his simple machines of 1936 could, in principle, compute anything that could be programmed. Given the right software, therefore, a powerful enough *universal* machine could mimic any computational procedure that could be taught explicitly by one person to another. And far more practical hardware was rapidly being developed – on both sides of the Atlantic – that could execute such algorithms with ever-increasing speed and reliability. A computer revolution could only be a matter of time.

But most people were, of course, quite unaware of this progress, and indeed deliberately kept ignorant of their immense debt to Bletchley Park and its machines. It was to be another three decades before computers would start to become widely familiar, and in the late 1940s the idea that an electronic machine could "think" in anything like the way that we do would have generally seemed preposterous, and even offensive. The British were still overwhelmingly Christian, believing in an immaterial soul that was the seat of our reason and consciousness, and which could survive destruction of the body. Some pressures on this consensus were beginning to show, especially amongst intellectuals aware of the Enlightenment challenges and of our evolutionary place in nature. But the manifest evils of war had, if anything, made religion more salient to those who had been bereaved, rather than generating widespread doubt about a benign creation. Not until the major social changes of the 1960s would the religious commitment of the British people start to be seriously undermined. Those changes included the repeal, in 1967, of the long-standing (and religiously-inspired) legal prohibition of male homosexuality, under which Turing was prosecuted in 1952 and which tragically led to his early death.

Against this background, Turing in the late 1940s began to argue the case for machine intelligence, with a particular focus on undermining what he saw as the dominant *prejudice* against computers. In 1947 he gave a lecture on the ACE to the London Mathematical Society, alluding to his 1936 results on the limitations of computers but emphasising explicitly – in his concluding paragraph – the idea of *fairness* in judging them:

> "It has … been shown with certain logical systems there can be no machine which will distinguish provable formulae of the system from unprovable. … Thus if a machine is made for this purpose it must in some cases fail to give an answer. On the other hand if a mathematician is confronted with such a problem he would search around a[nd] find new methods of proof … I would say that fair play must be given to the machine. Instead of it sometimes giving no answer we could arrange that it gives occasional wrong answers. But the human mathematician would likewise make blunders when trying out new techniques. It is easy for us to regard these blunders as not counting and give him another chance, but the machine would probably be allowed no mercy. … To continue my plea

> for 'fair play for the machines' when testing their I.Q. …, the machine must be allowed to have contact with human beings in order that it may adapt itself to their standards. The game of chess may perhaps be rather suitable for this purpose, as the moves of the machine's opponent will automatically provide this contact." (1947, pp. 393-4)

The following year, Turing completed a report for the Director of the National Physical Laboratory which was published only after his death. Entitled "Intelligent Machinery", it has been described by Jack Copeland as "the first manifesto of artificial intelligence" (2004, p. 401). The final numbered section, entitled "Intelligence as an emotional concept", runs as follows:

> "The extent to which we regard something as behaving in an intelligent manner is determined as much by our own state of mind and training as by the properties of the object under consideration. If we are able to explain and predict its behaviour or if there seems to be little underlying plan, we have little temptation to imagine intelligence. With the same object therefore it is possible that one man would consider it as intelligent and another would not; the second man would have found out the rules of its behaviour.
>
> It is possible to do a little experiment on these lines, even at the present stage of knowledge. It is not difficult to devise a paper machine which will play a not very bad game of chess. Now get three men as subjects for the experiment A, B, C. A and C are to be rather poor chess players, B is the operator who works the paper machine. … Two rooms are used with some arrangement for communicating moves, and a game is played between C and either A or the paper machine. C may find it quite difficult to tell which he is playing.
>
> (This is a rather idealized form of an experiment I have actually done.)" (1948, p. 431)

Here we have a clear anticipation of the Turing Test, applied to the particular case of chess. But notice how closely its setup is tied to the overcoming of *prejudice* – the extent to which our judgements of intelligence are subjectively dependent on our own assumptions, and even "emotional" in character. To guard against this, Turing's proposed experiment ensures fairness through a "blind" testing procedure, a procedure which he will later advocate explicitly as the appropriate method for making such judgements.[25]

Introducing the Imitation Game

Turing's famous *Mind* paper of 1950, "Computing Machinery and Intelligence", starts in a way that is both cavalier and rather confusing:

[25] Thus Turing's section heading need not be interpreted as saying that our judgements of intelligence *ought to be* made on an "emotional" basis, nor that intelligence is a *response-dependent* concept (as suggested by Proudfoot 2017, pp. 304-5). Turing's talk of our "temptation to imagine intelligence" suggests instead that when we ascribe intelligence, we take ourselves to be hypothesising something quite distinct from our own immediate feelings.

> "I propose to consider the question, 'Can machines think?' This should begin with definitions of the terms 'machine' and 'think'. The definitions might be framed so as to reflect so far as possible the normal use of the words, but this attitude is dangerous. If the meaning of the words 'machine' and 'think' are to be found by examining how they are commonly used it is difficult to escape the conclusion that the meaning and the answer to the question, 'Can machines think?' is to be sought in a statistical survey such as a Gallup poll. But this is absurd. Instead of attempting such a definition I shall replace the question by another, which is closely related to it and is expressed in relatively unambiguous words." (§1, p. 441)

One obvious objection is that although *the meanings of words* might plausibly be discovered from – or even determined by – their "normal use" (a view associated with Ludwig Wittgenstein, whom Turing knew well), that is very far from implying that *the answer to a question* framed in such words is to be found in a Gallup poll. Another obvious problem is with the whole notion of *replacing* one question by another: is this supposed to imply that the answer to the second can simply be carried over to the first? But how can this be guaranteed, since their meaning must presumably be different if the first question has ambiguities that are not reflected in the second? Turing never explains or even discusses these issues in respect of what he takes to be his two "closely related" questions. It is also somewhat ironic that his "relatively unambiguous" replacement question itself turns out to be highly ambiguous!

There is a great deal more vagueness, loose argument, and even humour to come throughout the paper, which falls far short of the rigour that would now be expected of a paper published in a leading philosophical journal. So when analysing it, we are well advised to bear in mind what Robin Gandy tells us about the context in which it was written:

> "The 1950 paper was intended not so much as a penetrating contribution to philosophy but as propaganda. Turing thought the time had come for philosophers and mathematicians and scientists to take seriously the fact that computers were not merely calculating engines but were capable of behaviour which must be accounted as intelligent; he sought to persuade people that this was so. He wrote this paper – unlike his mathematical papers – quickly and with enjoyment. I can remember him reading aloud to me some of the passages – always with a smile, sometimes with a giggle. Some of the discussions of the paper I have read load it with more significance than it was intended to bear." (Gandy 1996, p. 125)

This should counsel us against, for example, putting decisive weight on one individual passage that favours a very specific interpretation, when other passages point towards a different view. In some cases Turing's own position might not have been entirely clear, and sometimes he might simply have been careless in expressing it. In seeking to understand the Turing Test, therefore, literal reading of the text must be subject to the discretion of interpretative judgement.

Turing's new question – his replacement for "Can machines think?" – is set in the context of an "imitation game" involving a man A, a woman B, and an interrogator C – whose task is to identify, through written questions and answers, which of the other two (identified through the aliases "X" and "Y") is the man and which is the woman. The man's task is to deceive the interrogator into making the wrong identification by pretending to be a woman, while the woman attempts to assist the interrogator by being herself. In Turing's illustrative example, the interrogator asks "Will X please tell me the length of his or her hair?" – and X, who is actually the man, falsely answers "My hair is shingled, and the longest strands are about nine inches long" (§1, p. 441). Turing now imagines Y responding "I am the woman, don't listen to him!". (However, in later discussions of the game/test setup, the interrogations become one-to-one and interactions between the two "witnesses" disappear, as do the "X" and "Y" aliases.)

Understanding the Turing Test

The very short next paragraph – which ends §1 of the paper – introduces what we now know as the Turing Test. Since we shall be referring back to it in what follows, I mark it with "(*)":

> (*) "We now ask the question, 'What will happen when a machine takes the part of A in this game?' Will the interrogator decide wrongly as often when the game is played like this as he does when the game is played between a man and a woman? These questions replace our original, 'Can machines think?'" (§1, p. 441)

Turing had led us to expect one question to replace "Can machines think?", but now he offers two, though since the first of them does not have a yes/no answer, it seems most charitable to interpret that as mere framing for the second. More problematically, however, his description of this new game is seriously incomplete. Taken literally, it seems as though the machine is to be slipped into the game in place of the man, *attempting to imitate a woman*. But Turing says nothing to indicate whether or not the interrogator C is to be told of the change, yet if he or she is not told about it – and is left under the illusion that the other two participants are a man and a woman – then the supposed test for "thinking" becomes ludicrous.[26] Imagine, for example, what might happen if participant A, identified as X by the interrogator, now becomes a simple chatbot with a narrow range of answers tuned to the game context:

[26] Hayes and Ford (1995, p. 972), Sterrett (2000, pp. 542, 545-52), and Traiger (2000, pp. 565-7) all apparently see it as an advantage that the interrogator should be expecting the two "witnesses" to be a man and a woman, supposedly making the contest more demanding and subtle. Saygin *et al.* (2000, pp. 466-7), though less explicit on the interrogator's expectations, likewise favour a gender-based test on somewhat similar grounds. But these discussions all overlook the risk that this setup can, on the contrary, dramatically *lower* the bar for the computer, since there is then no requirement that it should succeed in passing itself off as an *intelligent* woman.

C: “Will X please tell me the length of his or her hair?”

A/X: “I want to you know that I am the woman.”

B/Y: “I am the woman – don’t listen to him.”

C: “OK, Y, please tell me the length of *your* hair.”

B/Y: “My hair is shingled, and the longest strands are about nine inches long.”

A/X: “He’s lying – I am the woman.”

As this goes on, we can expect a sequence of intelligent answers from the woman (B), and mere repetitive claims of womanhood from the chatbot (A), but since the interrogator is under the impression that the two participants are a man and a woman, he or she can only conclude that one of the two is obsessive or mentally defective, with no obvious clue as to whether the imbecile is male or female (since either way, the answers from the intelligent participant are likely to be relevantly similar).[27] Perhaps this will make it a coin-toss whether he or she guesses correctly as to which is the genuine woman – in which case the chatbot will presumably perform better in this game than a typical man – but that hardly amounts to a serious test for intelligence![28]

Thus the new form of Turing’s imitation game can only make reasonable sense if the interrogator is to be told that the decision is now between a human and a computer. But in that case, the question of gender becomes an irrelevant complicating factor: the interrogator is far more likely to focus on questions that attempt to distinguish person from machine, than questions that attempt to distinguish male from female (given that one of the participants is of neither sex). And indeed it is striking that in the rest of Turing’s paper, not a single one of the questions he proposes is gender-related: they concern skill at poetry, arithmetic, and chess, with no hint of gender relevance (see especially §2, p. 442; §6.4, p. 452; and §6.5, p. 454).

The subsequent text of the paper also repeatedly makes clear that Turing intends his test to be focused on the human/computer rather than female/male decision.[29] In §2, entitled “Critique of the New Problem” and starting *immediately* after passage (*) quoted above, Turing six times talks explicitly of a “man” – even implying that the machine’s obvious strategy is *to imitate a man* – and makes no mention whatever of women or the gender issue (pp. 442-3). A

[27] A more interesting chatbot in the same spirit might rant at length about the oppressive and binary nature of testing in general, and how testing for gender in particular is both patriarchal and problematically culture-relative, especially when focused on such trivia as length or styles of hair. Turing (1952, p. 495, quoted later) says that “the machine would be permitted all sorts of tricks”, and one possible trick here would be to parody an obsessive feminist who says a great deal, but without actually giving any detailed attention to any of the questions.

[28] I am assuming here that a typical woman will generally outperform a typical man in this gender imitation game, so that a man is doing extremely well if he achieves a 50% probability of fooling an interrogator. But of course it might be possible that a talented man could outperform most women, perhaps by taking advantage of cognitive biases on the part of the interrogator (such as false sexist assumptions about male and female interests or abilities).

[29] For more detailed textual discussion coming to the same conclusion, see Piccinini (2000).

similar pattern continues for the remainder of the paper, with women mentioned only in the context of an imagined "theological objection" (§6.1 p. 449), while the words "man" or "men" occur a further 30 times. Mostly these words appear to be used gender-neutrally (as was then standard), but this again suggests that gender is quite irrelevant to the intended Turing Test.

Turing describes his test more carefully at the end of §5 when, having explained the universality of digital computers – and clarified that his concern is not just with *current* computers, but with *imaginable* future technology – he rephrases the key question as follows:

> "It was suggested tentatively [at the end of §3] that the question, 'Can machines think?' should be replaced by 'Are there imaginable digital computers which would do well in the imitation game?' … But in view of the universality property we see that … [this question] … is equivalent to this, 'Let us fix our attention on one particular digital computer *C*. Is it true that by modifying this computer to have an adequate storage, suitably increasing its speed of action, and providing it with an appropriate programme, *C* can be made to play satisfactorily the part of A in the imitation game, *the part of B being taken by a man*?'" (§5, p. 448, emphasis added)

Since gender has not featured in the discussion since §1, it seems clear that "man" here should be interpreted gender-neutrally and the test interpreted accordingly, as comparing human against computer, which is indeed how the Turing Test has most commonly been interpreted.

Another change here concerns the criterion for computer success. Whereas (*) suggested a comparison with the success rate of a man attempting to imitate a woman – which now seems of little relevance to the question at issue – §5 asks instead whether the computer "can be made to play satisfactorily the part of [a human] in the imitation game".[30] Given the setup of the game, the most obvious way of assessing whether its performance is indeed "satisfactory" is in terms of the probability (judged by observed frequency) with which an interrogator is fooled into wrongly identifying which of the two "witnesses" is the human, and which the computer. This is exactly the criterion that Turing appears to adopt when at the beginning of §6 he makes his famous 50-year prediction:

> "I believe that in about fifty years' time it will be possible to programme computers … to make them play the imitation game so well that an average interrogator will not have more than 70 per cent. chance of making the right identification after five minutes of questioning." (§6, p. 449)

[30] Copeland and Proudfoot (2009, p. 124; cf. Copeland 2017, p. 271) take (*) more seriously, claiming that "the man-imitates-woman game is … part of the protocol for scoring the test". On their view, "If the computer (in the computer-imitates-human game) does no worse than the man (in the man-imitates-woman game), it passes the test." Against this, we have already seen evidence that (*) is carelessly written, and if Turing indeed takes the man-imitates-woman game to be providing an important baseline for assessment, one would reasonably expect him to give some consideration to how well a typical man would do in that game, which he never does.

Once we are in the domain of statistical frequency, however, there is no need for continuing with the three-participant game.[31] For it now makes more sense to conduct the exercise using a sequence of individual *viva-voce* examinations, a format which Turing adopts after the first section of the paper (at §2 p. 442 and §6.4 p. 452). His subsequent works also move in this direction. In the 1951 lecture "Can Digital Computers Think?", Turing makes no mention of the competitive game, but speaks instead of "something like a viva-voce examination, but with the questions and answers all typewritten" (p. 484). And in a 1952 BBC radio discussion, he adds more rigour by bringing in a panel of judges rather than a single interrogator, and emphasises that objectivity demands repeated tests with a mixed population of people and machines:

> "I would like to suggest a particular kind of *test* that one might apply to a machine. You might call it a test to see whether the machine thinks, but it would be better to avoid begging the question, and say that the machines that pass are (let's say) 'Grade A' machines. The idea of the test is that the machine has to try and pretend to be a man, by answering questions put to it, and it will only pass if the pretence is reasonably convincing. A considerable proportion of a jury, who should not be expert about machines, must be taken in by the pretence. They … [can] ask it questions, which are transmitted through to it: it sends back a typewritten answer … [questions can be about] anything. … the machine would be permitted all sorts of tricks so as to appear more man-like, such as waiting a bit before giving the answer, or making spelling mistakes … We had better suppose that each jury has to judge quite a number of times, and that sometimes they really are dealing with a man … That will prevent them saying 'It must be a machine' every time without proper consideration. Well, that's my test. … It's not the same as 'Do machines think', but it seems near enough for our present purpose, and raises much the same difficulties." (Turing *et al.*, 1952, p. 495)

It may seem odd to take a radio discussion rather than a paper in *Mind* as authoritative, but this seems to provide Turing's most considered presentation of the Turing Test.

Does Turing's "Intelligence" have to be Human-Like?

The Turing Test certainly gives a superficial impression of advocating a human-centred understanding of "thinking" or "intelligence" (terms that Turing tends to treat as equivalent).[32]

[31] Especially given that Turing never suggests that there might be interaction between the human and the computer of the kind that he described in the gendered game (e.g. "Don't listen to that machine – I'm the human!").

[32] Turing's (1947) and (1948), aimed at mathematical audiences, refer repeatedly to *intelligence* rather than *thinking*, and although the (1948) alludes to "our task of building a 'thinking machine'" (p. 420), the inverted commas suggest that he considers the term colloquial rather than precise. This diagnosis is confirmed by the 1950 paper and the radio broadcast (1951a), which both refer to *intelligence* in their title, yet both start with a quotation about *thinking*. And although machine *thought* is mentioned far more often in the 1950 paper that in the other pieces, it is striking that nearly every such reference is either explicitly quoted or implicitly put in the voice of an objector, while even the two exceptions (at §2, p. 442 and §6, p. 449) are indirect in nature. It therefore seems reasonable to conclude that Turing views *intelligence* as the more correct term, with *thought* as a colloquial alternative.

Some influential interpreters have even seen it as intended to provide an "operational definition",[33] but this goes too far. Turing's very first sentence in the 1952 BBC discussion is "I don't want to give a definition of thinking" (p. 494). And in the 1950 paper itself, he seems explicitly to allow that machine intelligence could be very different from the human variety, thus implying that passing of the Turing Test is not *necessary* for machine intelligence:

> "May not machines carry out something which ought to be described as thinking but which is very different from what a man does? This objection is a very strong one, but at least we can say that if, nevertheless, a machine can be constructed to play the imitation game satisfactorily, we need not be troubled by this objection." (§2, p. 442)

Whether passing the test is *sufficient* to deserve the accolade of intelligence may depend on where the threshold is drawn. As we saw above, Turing's 1952 discussion is relatively guarded, suggesting we "say that the machines that pass are (let's say) 'Grade A' machines", and combining this with the vague threshold that they "only pass if the pretence is reasonably convincing" (p. 495). The 1950 paper might seem to imply a more precise threshold, whereby computers need to "play the imitation game so well that an average interrogator will not have more than 70 per cent. chance of making the right identification after five minutes of questioning" (§6, p. 449). But this is merely a *prediction* of what Turing believes will be possible "in about fifty years' time", and he nowhere commits himself to its significance.

For an example of what Turing apparently considers to be a *successful* test, we must look to the *viva-voce* dialogue in §6.4 of the 1950 paper, where he is responding to what he calls "The Argument from Consciousness":

> "Interrogator: In the first line of your sonnet which reads 'Shall I compare thee to a summer's day', would not 'a spring day' do as well or better?
>
> Witness: It wouldn't scan.
>
> Interrogator: How about 'a winter's day'. That would scan all right.
>
> Witness: Yes, but nobody wants to be compared to a winter's day.
>
> Interrogator: Would you say Mr. Pickwick reminded you of Christmas?
>
> Witness: In a way.
>
> Interrogator: Yet Christmas is a winter's day, and I do not think Mr. Pickwick would mind the comparison.
>
> Witness: I don't think you're serious. By a winter's day one means a typical winter's day, rather than a special one like Christmas." (p. 452)

[33] For example Shannon and McCarthy (1956), p. v; Miller (1973), p. 595; Hodges (1983), p. 415; French (1996), pp. 11-12; Michie (1996), p. 29.

Ignoring for the moment the context of Turing's (tendentious) response to the question of consciousness,[34] let us appreciate the force of his implicit point, that *if we encountered a machine capable of responding with this degree of sophistication, in response to comparably difficult questions chosen across a wide range of subjects by some independent group of interrogators, then it would seem unreasonable to withhold the attribution of "intelligence" to that machine*. If this is accepted, then the Turing Test provides a sort of existence proof of possibility. For here is a *conceivable* outcome that should be enough to persuade us of a machine's intelligence, and an outcome, moreover, of whose genuine *possibility* Turing is entirely confident.

Turing's confidence stems, of course, from his 1936 work on the *universality* of digital computers, a notion which he emphasises strongly in both the 1950 paper (§5, pp. 446-8) and his 1951 radio lecture (pp. 482-4).[35] This absolves him from any obligation to analyse the specific powers of any particular computer, given that:

> "we are not asking … whether the computers at present available would do well, but whether there are imaginable computers which would do well" (§3, p. 443)

If this is indeed Turing's primary question, then there is no need for him to be very specific about the level of performance required before he is prepared to commit himself to saying that a machine is genuinely "thinking" or "intelligent". It is enough to have an existence proof of a potential *future* machine that would *clearly* pass: a potential to be fulfilled only when digital hardware and programming techniques have caught up with the theoretical possibility.

From this perspective, however, there need after all be nothing very special about human-likeness in Turing's theoretical vision. Once we have a universal computer, *any* theoretically feasible level or flavour of algorithmic "thinking" is potentially achievable, given appropriate memory capacity, speed, and programming, and within the bounds of possibility explored in his 1936 paper. So it seems that the crucial role of the Turing Test need not be to provide any realistic benchmark of practical progress in AI, but instead, to serve as the context for an *extreme exemplar* that throws down a gauntlet by facing critics of AI with an imagined level of performance that any fair-minded interrogator – forced to judge on that performance alone –

[34] Turing's example conversation concerns poetry, because he is responding to Geoffrey Jefferson's Lister Oration for 1949, as quoted at §6.4, p. 451: "Not until a machine can write a sonnet … because of thoughts and emotions felt, and not by the chance fall of symbols, could we agree that machine equals brain". Jefferson here poses false alternatives, and Turing's reply on p. 452 addresses only the second of these: his machine's *viva-voce* responses are clearly better than *chance*, but this does nothing to prove that there is genuine *emotion* behind the words.

[35] Such "universality" seems to encompass both the Church-Turing Thesis and the possibility of a Universal Turing Machine. Also relevant here is Turing's argument that a discrete-state machine can in principle mimic a continuous machine – such as the nervous system appears to be – with arbitrary precision (§6.7, pp. 456-7).

would have to count as "intelligent". After the critics have been won over to the extent of acknowledging this as a *hypothetical* possibility, the universality of digital computers can then lead them on to acknowledge intelligent machinery as a *genuine* possibility.

Reconsidering Standard Objections to the Turing Test

This approach accordingly blunts one of the most oft-repeated objections to the Turing Test, that of alleged *anthropomorphism*. It can also provide a response to what we might call *the chatbot objection*, that the level of performance suggested by Turing's 50-year prediction – namely fooling an "average interrogator" 30% of the time "after five minutes of questioning" – may well be relatively easy to achieve using a machine that is very far from intelligent. This objection is fatal to the Turing Test if it is conceived as *a measure of progress towards AI* or as *a way of gauging the relative "intelligence" of individual programs*. And the main problem here is a different kind of human-centredness, not in respect of what is being measured (namely, the desired human-like reactions of the program), but rather, in respect of who is doing the measuring (namely, a naturally biased and gullible human judge). Joseph Weizenbaum's *ELIZA* program of 1966 taught us something previously unexpected, which would probably have been as surprising to Turing as to others: namely, that an extremely crude "chatbot" program, in conversation with a naïve interlocutor, is often able to sustain a credible conversation for at least a few interchanges. I have discussed this issue at greater length elsewhere,[36] but for present purposes it is enough to point out that advocates of the "extreme exemplar" conception of the Turing Test proposed above would be quite free to take on the lesson of *ELIZA*. Accordingly, they can simply concede that the 50-year prediction has turned out to be too easily fulfilled – by crude syntactic tricks and misdirection rather than by sophisticated information processing – to provide any useful benchmark of intelligence. But the far more demanding standard of Turing's *viva-voce* example remains entirely untouched by this concession.

We can also rebut what might be called the *lookup table* objection, that a system based on a giant lookup table of responses could in principle pass the Turing Test to any desired degree of sophistication.[37] To my mind this objection exemplifies a dubious use of that philosophers' term "in principle", for even to set up appropriate responses for a few short exchanges would

[36] See Millican (2013), pp. 596-8. For a system which incorporates a faithful emulation of the original *ELIZA*, enabling the internal mechanisms to be inspected in real time, see http://www.philocomp.net/ai/elizabeth.htm.

[37] This objection dates back to Shannon and McCarthy (1956, p. vi), but is most associated with Block (1981), perhaps owing to the happy coincidence that "Blockhead" is a remarkably apt name for the postulated program. Another objection in a similar spirit is Searle's notorious Chinese Room Argument (1984), which likewise attempts to pump our intuitions in an imagined situation of outrageous unfeasibility (see Millican 2013, pp. 588-90).

require more memory cells than there are atoms in the visible universe, and the numbers grow exponentially with every new exchange. Trusting our intuitive judgements in such a fairy-tale scenario seems very questionable, and the most that can be expected from it is to persuade us that intelligence cannot plausibly be *defined* in terms of the Turing Test. But the objection has no force against the "extreme exemplar" view, whose point is to claim that the sort of system Turing envisages is genuinely feasible within the not-too-distant future. That claim is, of course, potentially disputable, as Turing himself recognises. In their 1952 radio discussion, Newman talks of the Manchester machine requiring "thousands of millions of years" to analyse a game of chess by brute force, and says that to suppose such things "will be done in a flash on machines of the future, is to move into the realms of science fiction". Turing responds: "If one didn't know already that these things [such as playing chess well] can be done by brains within a reasonable time one might think it hopeless to try with a machine. The fact that a brain *can* do it seems to suggest that the difficulties may not really be so bad as they now seem." (Turing *et al.* 1952, pp. 503-4). Advances in software technology since then indeed provide support for this style of response. As one topical example, computer chess programs now routinely defeat grandmasters, even running on commonplace hardware. And as another (very different) example, deep learning systems have in recent years proved able to solve subtle pattern-recognition problems that previously seemed intractable.[38] As one possible application of these, it now looks relatively plausible that such systems could enable machines to mimic the sort of culturally nuanced "subcognitive" reactions highlighted by Robert French in his attack on the Turing Test, which seemed to many at the time to be resistant to any foreseeable algorithmic simulation.[39]

One important type of objection to the Turing Test remains, based on the general idea that behavioural indistinguishability in terms of answers to questions cannot prove similarity in terms of subjective experience. The premise here is entirely correct, since – to put the point

[38] Neural networks might themselves be considered liable to the lookup table objection, on the grounds that they operate without any "intelligence", and indeed are rather like fuzzy lookup tables. But such a network could at most play a subsidiary role within any system that aspired to pass the Turing Test, given the multitude of different kinds of problem – many requiring specific and fine-grained answers – that could be set, through exponentially many possible sequences of questions (e.g. about arithmetic, or chess, or poetry, or any of a thousand other subjects). Part of the genius of Turing's choice of test is that to pass it in full generality, any practically feasible program would have to be capable of operating with great precision and discrimination over a wealth of coordinated data structures.

[39] See, for example, the "rating games" in French (1996, pp. 18-22), which include asking the subject of the Turing Test to rate "Flugly" as the name of a teddy bear or as the name of a glamorous female movie star. Note also that on the "extreme exemplar" conception of the Turing Test, one can plausibly reject the whole idea that simulation of highly culture-relative judgements should be considered a requirement. For on this conception, the point of the Test is enable presentation of an exemplar that would unequivocally count as intelligent on any reasonably fair standard, *with indistinguishability from a human playing a procedural rather than normative role, facilitating the assurance of fairness through "blind" judging*. But demanding absolute similarity in respect of cultural judgements is *not* fair, as becomes obvious if we imagine applying such a test to humans from different cultures to our own.

crudely – if we know that a system has been programmed to generate the relevant responses through the execution of some algorithm, then the occurrence of those responses cannot give us evidence of some *other* cause. So if our own responses are in fact generated (at least in part) by subjective, conscious experience, then a program reproducing that external behaviour *without consciousness being causally involved* cannot be bringing it about in an identical manner; and hence any argument from similar output to similar causation is completely undermined.

Especially in these days of inscrutable machine learning, it is important to emphasise that this objection does not depend on our having precise understanding of the algorithm responsible for a machine's behaviour. It is enough to know in general that the algorithms are designed to operate by standard computational methods, on hardware systems whose behaviour is well understood in terms of physical processes that have no reliance on consciousness. In these circumstances, there is no basis whatever for supposing that consciousness somehow magically makes an entrance once a certain kind of behaviour is produced, when that behaviour is already sufficiently accounted for by the algorithmic implementation.

A potential response to this sort of objection, in the spirit of Turing's "fair play for the machines", is to suggest that on similar principles, if we understood human neurophysiology and biochemistry well enough, then we would be able to explain human behaviour entirely in *those* terms, thereby "proving" that consciousness plays no role in *human* behaviour.[40] But such a response is implicitly taking for granted exactly what the proponent of the objection will deny, namely, that consciousness is indeed causally inert in human behaviour (or at best, that it is a mere abstraction from behaviour and functional role, rather than something ontologically distinct). And it is important to note here that such a denial need not be founded at all on some sort of Cartesian dualism or belief in souls; for it is entirely compatible with accepting, on the basis of evolutionary evidence, that consciousness is almost certainly a function of physical matter. *How on earth consciousness arose,* and *how it can be generated* by neurophysiology and biochemistry (etc.) is currently a mystery, and the relevant sciences might well have to go through conceptual revolutions – as did the physical sciences – before we can even glimpse solutions to their most fundamental questions. But *that consciousness somehow arose*, and *that it does play a genuine causal role in our own behaviour*, seem as obvious as almost anything can be. When we have only started to scale the foothills of these sciences relatively recently, it is hubristic to suppose that we can predict what their ultimate form will be, and absurdly so to assume in advance that this ultimate form can give no genuine causal role to consciousness.

[40] I am grateful to an anonymous referee for posing this very response.

Turing's own treatment of "The Argument from Consciousness" obscures this crucial epistemological asymmetry between the causation of human and machine behaviour by raising the spectre of solipsism, with a touch of humour:

> "According to the most extreme form of this view the only way by which one could be sure that a machine thinks is to *be* the machine … Likewise according to this view the only way to know that a *man* thinks is to be that particular man. It is in fact the solipsist point of view. It may be the most logical view to hold but it makes communication of ideas difficult. A is liable to believe 'A thinks but B does not' whilst B believes 'B thinks but A does not'. Instead of arguing continually over this point it is usual to have the polite convention that everyone thinks." (§6.4, p. 452)

It may be that an extreme solipsist would indeed view the consciousness of another person and of a machine with equal scepticism, but this is no basis on which to draw a balanced and rational conclusion. The common-sense position is instead to acknowledge that one's own consciousness is very probably indicative of other people's consciousness also, given their similar biological origin and nature. But again, this gives *no ground whatever* for extrapolating consciousness to a machine, however similar its behaviour, if we have reason to believe that the similarity of that behaviour is in no way driven by such biological processes, but instead by some program designed for the purpose (whose operations themselves require no machine consciousness).

This is the weakest major point in Turing's two seminal papers, where he should have been prepared to break away from the human-centred paradigm of "intelligence" that he had strategically highlighted in his famous Test. Here he should have had the courage to say that a machine capable of giving comparable answers to those of an expert human would be a clear exemplar of intelligence *whether or not it was conscious*. Intelligence is standardly understood to be a measure of sophisticated information processing for some purpose, not a measure of subjective experience. Human manifestations of intelligence may indeed be commonly – perhaps usually – accompanied by subjectivity. But after Turing has shown that sophisticated information processing is something that can equally be achieved by a machine (and without invoking any subjective experience), it then becomes entirely appropriate to distinguish the information processing from the subjectivity, and to reserve the word "intelligence" for the former.[41] This would involve some revision of our naïve conceptual scheme, away from a human-centred view of intelligence. But such revision, Turing should have insisted, would be fully justified in the light of his own fundamental discoveries.[*]

[41] For more extensive discussion of these issues, which are merely sketched here, see Millican (2013, pp. 590-6).

[*] I am grateful to Andrew Hodges, Alexander Paseau, Shashvat Shukla, John Truss, and Stan Wainer for helpful discussion when preparing this paper, and also two anonymous referees.

Bibliography

Block, Ned (1981), "Psychologism and Behaviorism", *Philosophical Review* 90, pp. 5-43.

Champernowne, D. G. (1933), "The Construction of Decimals Normal in the Scale of Ten", *Journal of the London Mathematical Society* 8, pp. 254-60.

Church, Alonzo (1934), "The Richard Paradox", *American Mathematical Monthly* 41, pp. 356-61.

Cooper, S. Barry and Jan van Leeuwen, eds (2013), *Alan Turing: His Work and Impact*, Waltham Massachusetts: Elsevier.

Copeland, B. Jack, ed. (2004), *The Essential Turing*, Oxford: Clarendon Press.

Copeland, B. Jack (2017), "Intelligent Machinery", in Copeland *et al*. (2017), pp. 265-75.

Copeland, Jack and Diane Proudfoot (2009), "Turing's Test: A Philosophical and Historical Guide", in Epstein *et al.* (2009), pp. 119-38.

Copeland, B. Jack, Jonathan P. Bowen, Mark Sprevak, and Robin Wilson, eds (2017), *The Turing Guide*, Oxford: Oxford University Press.

Robert Epstein, Gary Roberts, and Grace Beber, eds (2009), *Parsing the Turing Test*, Dordrecht: Springer.

Fan, Zhao (2020), "Hobson's Conception of Definable Numbers", *History and Philosophy of Logic* 41, pp. 128-39.

French, Robert M. (1990), "Subcognition and the Limits of the Turing Test", *Mind* 99, pp. 53-65 and reprinted in Millican and Clark (1996), pp. 11-26.

Gandy, Robin (1988), "The Confluence of Ideas in 1936", in Herken (1988), pp. 55-111.

Gandy, Robin (1996), "Human versus Mechanical Intelligence", in Millican and Clark (1996), pp. 125-36.

Gödel, Kurt (1962), *On Formally Undecidable Propositions of Principia Mathematica and Related Systems*, tr. B. Meltzer with an introduction by R. B. Braithwaite, New York: Basic Books.

Hayes, Patrick and Kenneth Ford (1995), "Turing Test Considered Harmful", *IJCAI*-95, pp. 972-7.

Herken, Rolf, ed. (1988), *The Universal Turing Machine: A Half-Century Survey*, Oxford: Oxford University Press.

Hilbert, David and Wilhelm Ackermann (1928), *Grundzüge der theoretischen Logik*, Berlin: Springer.

Hobson, E. W. (1921), *The Theory of Functions of a Real Variable and the Theory of Fourier's Series*, second edition, Cambridge: Cambridge University Press.

Hodges, Andrew (1983), *Alan Turing: The Enigma of Intelligence*, London: Burnett.

Hodges, Andrew (1988), "Alan Turing and the Turing Machine", in Herken (1988), pp. 3-15.

Hodges, Andrew (2009), "Alan Turing and the Turing Test", in Epstein *et al.* (2009), pp. 13-22.

Hodges, Andrew (2013), "Computable Numbers and Normal Numbers", in Cooper and van Leeuwen (2013), pp. 403-4. This is immediately followed by Turing's "A Note on Normal Numbers" (pp. 405-7) and Verónica Becher's "Turing's Note on Normal Numbers" (pp. 408-12).

Michie, Donald (1993), "Turing's Test and Conscious Thought", *Artificial Intelligence* 60, pp. 1-22 and reprinted in Millican and Clark (1996), pp. 27-51.

Miller, P. H. (1973), “On the Point of the Imitation Game”, *Mind* 82, pp. 595-7.

Millican, Peter (2013), “The Philosophical Significance of the Turing Machine and the Turing Test”, in Cooper and van Leeuwen (2013), pp. 587-601.

Millican, Peter and Andy Clark (1996) eds, *Machines and Thought*, Oxford: Oxford University Press.

Petzold, Charles (2008), *The Annotated Turing: A guided tour through Alan Turing’s historic paper on computability and the Turing machine*, Indianapolis: Wiley.

Piccinini, Gualtiero (2000), “Turing’s Rules for the Imitation Game”, *Minds and Machines* 10, pp. 573-82.

Proudfoot, Diane (2017), “Turing’s Concept of Intelligence”, in Copeland *et al*. (2017), pp. 301-7.

Russell, Bertrand (2014), ed. Gregory H. Moore, *Collected Papers, Volume 5: Toward Principia Mathematica, 1905-08*, Abingdon: Routledge.

Saygin, Ayse Pinar, Ilyas Cicekli, and Varol Akman (2000), “Turing Test: 50 Years Later”, *Minds and Machines* 10, pp. 463-518.

Searle, John (1984), *Minds, Brains, and Science*, Cambridge Massachusetts: Harvard University Press.

Shannon, C. E. and J. McCarthy (1956), *Automata Studies*, Princeton: Princeton University Press.

Sprevak, Mark (2017), “Turing’s Model of the Mind”, in Copeland *et al*. (2017), pp. 277-85.

Sterrett, Susan G. (2000), “Turing’s Two Tests for Intelligence”, *Minds and Machines* 10, pp. 541-59.

Traiger, Saul (2000), “Making the Right Identification in the Turing Test”, *Minds and Machines* 10, pp. 561-72.

Turing, Alan M. (1936), “On Computable Numbers, with an Application to the *Entscheidungsproblem*”, *Proceedings of the London Mathematical Society*, Second Series, Vol. 42 (1936-7), pp. 230-65; reprinted in Copeland (2004), pp. 58-90 (page references are to this reprint).

Turing, Alan M. (1947), *Lecture on the Automatic Computing Engine*, delivered on 20 February 1947 to the London Mathematical Society, reprinted in Copeland (2004), pp. 378-94.

Turing, Alan M. (1948), *Intelligent Machinery*, report prepared for Sir Charles Darwin, Director of the National Physical Laboratory, reprinted in Copeland (2004), pp. 410-32.

Turing, Alan M. (1950), “Computing Machinery and Intelligence”, *Mind* 59, pp. 433-60; reprinted in Copeland (2004), pp. 441-64 (page references are to this reprint).

Turing, Alan M. (1951a), “Intelligent Machinery, A Heretical Theory”, talk for *The ’51 Society* broadcast on the BBC, typescript transcribed in Copeland (2004), pp. 472-5.

Turing, Alan M. (1951b), “Can Digital Computers Think?”, lecture broadcast on the BBC on 15 May 1951, transcribed in Copeland (2004), pp. 482-6.

Turing, Alan M., Richard Braithwaite, Geoffrey Jefferson, and Max Newman (1952), “Can Automatic Calculating Machines be Said to Think?”, discussion broadcast on the BBC on 10 January 1952, transcribed in Copeland (2004), pp. 494-506.

Weizenbaum, Joseph (1966), “ELIZA – A Computer Program For the Study of Natural Language Communication Between Man And Machine”, *Communications of the ACM* 9, pp. 36-45.

Whitehead, Alfred North and Bertrand Russell (1927), *Principia Mathematica*, second edition, Cambridge: Cambridge University Press. The relevant material is reprinted in the abridged version *Principia Mathematica to *56*, published by Cambridge in 1962.